الشرح

MIDTERM REVISION



Programming with C++

Faculty of Engineering – Tanta University
Electric Department
1st year



نخش في الموضوع علطول ☺☺

مقدمة نظرية
تقسيم لغات البرمجة

لغات البرمجة المنخفضة المستوى Low-Level Languages:

هي الجيل الأول من لغات البرمجة وهي لغة قريبة من لغة الآلة (0, 1) وتتعامل مع مكونات الكمبيوتر مباشرةً.

خواصها:

صعوبة الاستخدام. - صعوبة اكتشاف الأخطاء. - لا تعمل على حواسيب مختلفة.

أمثلة:

لغة الآلة Machine Language: تسمى اللغة الثنائية حيث إنها تتكون من سلسلة من 0 و1، وهي اللغة الوحيدة التي يفهمها الحاسب الآلي، حيث تحول جميع اللغات إلى لغة الآلة، حتى تتمكن معدات الحاسب الآلي من التفاهم معها.

لغة التجميعAssembly : استخدمت لغة التجميع لتسهيل البرمجة، فهي لغة الحروف المجمعة. وهي لغة تختصر بعض العبارات والرموز المستخدمة. ففيها يتم استبدال الرموز الرقمية في لغة الآلة بمجموعة من الكلمات الرمزية "المختصرة " باستخدام اللغة الإنجليزية. إذ من السهل نوعا التعامل معها، مثل:

L *for* Load, A *for* Add, B *for* Brave, *and* C *for* comp

By: Mhmd Darwish

لغات البرمجة عالية المستوى
High-Level Languages:

سميت بهذا الاسم لأنه أصبح بإمكان المبرمج كتابة البرنامج دون معرفة تفاصيل كيفية قيام الحاسب بهذه العمليات، كمواقع التخزين وتفاصيل الجهاز الدقيقة، وتعبيرات لغات المستوى العالي هي تعبيرات شبيهة إلى درجة كبيرة باللغة الطبيعية التي يستخدمها الإنسان في حياته للتواصل، والتخاطب مع الآخرين.

خواصها:

- غير مرتبطة بجهاز معين (أي يمكننا تنفيذ البرنامج المكتوب بلغة من لغات المستوى العالي على أكثر من جهاز).

- سهولة اكتشاف الأخطاء وتتبعها.

- سهولة قراءة الكود البرمجي والتعديل عليه.

أمثلة: Basic, Fortran, C, C++, C#, Java, Pascal, Cobol, Ruby, Python, ...

By: Mhmd Darwish

الفرق بين Compiler , Interpreter

الـ Complier: يقوم بفحص الكود المكتوب بلغة البرمجة كاملاً ثم يقوم بتنفيذه دفعة واحدة.

الـ Interpreter: نفس عمل الـ Complier ولكنه يختلف عنه في أنه يقوم بتنفيذ البرنامج سطر بسطر.

طرق كتابة البرنامج:

- Flow Chart.
- Pseudo Code. (Algorithm)
- Programming language Code.

By: Mhmd Darwish

خرائط التدفق Flow Charts

نستخدم هذه الطريقة في كتابة البرامج بدون التقيد بقواعد لغات البرمجة ويتم تحويلها إلى كود برمجي مكتوب بأي لغة برمجية ويستخدم فيها بعض الاشكال المتفق عليها لتمثيل خطوات معينة من بداية خوارزمية إلى نهايتها.

مميزات استخدام هذه الطريقة:

- تعطي صورة واضحة وكاملة للخطوات المطلوبة لحل مسألة معينة في ذهن المبرمج بحيث تساعده في الاطلاع على جميع اجزاء المسألة قبل تنفيذها.
- تبين للمبرمج الأخطاء في البرامج وبخاصة الأخطاء المنطقية والتي يعتمد اكتشافها على وضع التسلسل المنطقي لخطوات حل المسألة لدى المبرمج.
- تساعد المبرمج على ادخال اي تعديلات قد يحتاجها على اي جزء دون الحاجة لدراسة جميع اجزاء المسألة.
- تسهل على المبرمج فهم المسألة المعقدة والتي تكثر فيها الاحتمالات والتفرعات وبذلك تظهر الخريطة الخطوات الرئيسية بوضوح.

By: Mhmd Darwish

Flow Charts خرائط التدفق
الأشكال المستخدمة

الرمز	الحدث الذي يمثله	مثال
(شكل بيضاوي)	حدث طرفي Terminal لبيان بدء (Start) أو انتهاء (Stop) خريطة سير العمليات	Start / Stop
(مستطيل)	عملية حسابية (Process)	LET x+y
(متوازي أضلاع)	إدخال / إخراج INPUT \ OUTPUT ادخال البيانات / اخراج معلومات من والى الحاسب	PRINT I / INPUT x,y
(معين)	اتخاذ قرار Decision	NO y=x YES
(أسهم)	اتجاه تتابع العمليات Flow Line	(أسهم)
(شكل سداسي)	تكرار أو دوران Loop	For I = 1 to 5



By: Mhmd Darwish

Programming with C++
Headers

الــ Compiler مقســم إلى مجموعــة مــن الأقســام تســمى Headers ولا يتم استدعاءه مرة واحدة ولكن يتم استدعاء الــ Header المطلوب في البرنامج.

أمثلة:

<iostream> , <algorithm>, <cstdlib>, <math>, …

كل Header بيحتوي على مجموعة معينة من الأكواد مخصصة لأداء وظيفة في البرنامج. مثلا الهيدر <iostream> #include دة مسئول عن أوامر الـ cin والـ cout ... وهكذا.

By: Mhmd Darwish

C++ Components

Alphabetic:

A-Z Capital Letters. Or a – z Small Letters.

Digits (Numbers):

1 2 3 4 5 6 7 8 9

Digit → (خانة واحدة) رقم, Number → (أكثر من خانة) عدد

Reserved Words:

الكلمات المحجوزة هي مجموعة من الكلمات الغير متاح استخدامها في تعريف متغيرات البرنامج مثل: int, float, if, char, double, cin, cout, return, …

وهتلاحظ وانت بتكتب الكود إن الكلمات دى بتتلون بلون معين دليل على إنها محجوزة.

By: Mhmd Darwish

C++ Components
Special Characters:

/	Forward slash
\	Back slash
" ", ' '	Double and single quotes
:	Colon
;	Semi colon
,	Comma
#	Directive
>>	Input
<<	Output
( )	Parentheses
[ ]	Square Brackets
{ }	Braces

By: Mhmd Darwish

Comments in C++

التعليقات هي عبارة عن جمل يكتبها المبرمج عند كتابة البرنامج لشرح نقطة معينة أو وصف البرنامج وهذه الجمل يتجاهلها الـ Complier عند تنفيذ البرنامج.

لاحظ: التعليقات لا يراها سوى المبرمج. طب لازمتها إيه:

نفترض مثلاً إن انت بتعمل برنامج كبير وصل معاك مثلاً لـ ١٠٠ صفحة. فالـ comments دي مهمة جداً لإنها بتوضح كل مجموعة من الأسطر مثلاً مسئولة عن إيه.

By: Mhmd Darwish

Comments in C++
Comment Types

Single line comment:

ودة بيتكتب في سطر واحد وبيمثل بـ:

```
// write any comment
```

Multi line comment:

ودة ممكن كتابته في أكثر من سطر وبيمثل بـ:

```
/* line 1
Line 2
Line ..   */
```

Special Note:

ممكن تستخدم الـ comment عشان تلغي أي جزء من الكود إنت مش عايزه يتنفذ.

By: Mhmd Darwish

بنية البرنامج في C++

```
#include <iostream>
using namespace std;

// small program to print a word

main()
{
    cout << "Hello, my name is Darwish";
}

```

استدعاء الـ header الخاص بـ Input / Output

Compiler Header

Single line comment

Body Statement

Output

Hello, my name is Darwish

COLLEGETanta
م الآخر

By: Mhmd Darwish

Note

البرنامج السابق دة برنامج بسيط جداً لطباعة جملة عـلى الشاشـة ومكوناته:

- استدعاء الـ header iostream المسئول عن أوامر الإدخـال والإخـراج. (فيـه headers كتيرة بيتم استدعائها سيتم دراستها فـيما بعـد). ولاحـظ اسـتخدام الرموز < ,> ,#
- تعليـق مكـون مـن سـطر واحـد باسـتخدام الرمـز //. ولاحـظ عنـد التنفيـذ البرنامج نفض للـ comment.
- استخدام الدالة main( ) وسيتم شرحها مع الدوال فيما بعد.
- استخدام الكلمة المحجوزة cout والرموز >> وعلامتي التنصيص المزدوجة " " .

برنامج يقوم بحساب مساحة دائرة بمعلومية نصف القطر وطباعة الناتج على الشاشة.

```cpp
#include <iostream>
using namespace std;

// small program to calcualate the area of a circle

main()
{
    int A, R;  // or int A, R = 10
    float pi = 3.14;

    R = 10;
    A = pi * R * R;

    cout<< "Radius= " << R << " cm \n" << "Area= " << A << " cm square"; // "\n" provides a new line
}
```

Output

```
Radius= 10 cm
Area= 314 cm square
```


By: Mhmd Darwish

Note

- int: هو نوع من Data Types بيدل على الأعداد الصحيحة بدون علامـة عشريـة . أمـا float بيدل على الأعداد وع وجود علامة عشرية.

12 → int , 12.2 → float , 12.0 → float

- تستخدم العلامة ; للفصل بين كل تعبير **expression** وآخر.
- كل ما بداخل " " يعتبر نص حتى لو كان رقم. فمثلاً لو كتبـت "12" كـدة يبقـى 12 دة بيتعتبر نص يعني ملوش خواص الأرقام (مينفعش تجمعه أو تطرحه .. وهكذا).
- عند الطباعة يكتب النص بين " " ولكن تُكتب الأرقام والمتغيرات مباشرةً.
- يستخدم الأمر "n/" لإعطاء سطر جديد (يعني كل اللى مكتوب بعـده هيبقـى في سـطر جديد).
- جوة الـ cout يستخدم العلامة >> للفصل بين العناصر وبعضها.

By: Mhmd Darwish

cout

The standard output stream (cout):

تستخدم cout للطباعة على الشاشة.

Form:

```
cout << var_name << "Text" ;
```

- تستخدم العلامة << للفصل بين كل stream insertion وآخر.
- أي نص يُراد طباعته يوضع بين علامتي تنصيص " ".

Example:

```
int number = 10;
cout << number << " is my favorite number" ;
```

Output → **10 is my favorite number**

By: Mhmd Darwish

cin

The standard input stream (cin):

تستخدم cin لتخزين قيمة (يدخلها المستخدم) في متغير.

Form:

```
cin >> var;
```

- تستخدم العلامة >> للفصل بين كل stream extraction وآخر.

Example:

```
int number1, number2;
cin >> number1 >> number2;
cout << number1 << number2;
```

في المثال دة أول ما البرنامج يتنفذ هتلاقي الـ cursor | مستني إنك تدخله قيمة من الكيبورد وبعدين تضغط Enter عشان يخزن القيمة ويكمل تنفيذ البرنامج.

Conversation Method
طريقة المحادثة

الطريقة دي بتعتمد على مخاطبة الـ user لإعطاءه أمر معين أو ملاحظة معينة وفيها بيستخدم أمر cout والـ Escape Sequences زي \n, \t, \a, ...

مثال:

انت عامل برنامج ومطلوب فيه من المستخدم إنه يدخل ٣ أرقام صحيحة موجبة.

الطريقة الأمثل هي طريقة المحادثة:

```
int n1, n2, n3;
cout << "Ener three positive integers: \n";  //التخاطب مع المستخدم
cin >> n1 >> n2 >> n3;
```


By: Mhmd Darwish

انتبه !!

عند إدخال البيانات يجب مراعاة:

١. عدد البيانات. لازم تعرف المستخدم عدد البيانات اللي هيدخلها.. فمثلاً لو هتخزن ٣ قيم في ٣ متغيرات ممكن تقوله جوة الـ cout:
"Enter three numbers: "

٢. نوع المتغير اللي داخلاله البيانات يعني مثلا لو int تطلب من المستخدم إنه يدخل عدد صحيح ولو string تطلب من المستخدم إنه يدخل نص .. وهكذا.

٣. ترتيب المتغيرات. يعني مثلا لو كاتب `cin >> a >> b >> c;` يبقى المستخدم هيدخل قيمة a ثم b ثم c.

By: Mhmd Darwish

Escape Sequences

الـ Escape Sequences هـي مجموعـة مـن الحـروف المسـبوقة بعلامـة \ لعمـل بعـض التأثيرات والتنسيقات كعرض سطر جديد وإعطاء Tab .. وغيرها.

Value	Escape sequence
newline	\n
horizontal tab	\t
vertical tab	\v
backspace	\b
carriage return	\r
form feed	\f
alert	\a
backslash	\\
question mark	? or \?
single quote	'\
double quote	"\
the null character	٠\
octal	\ooo
hexadecimal	\xhhh
Unicode (UTF-8)	\uxxxx
Unicode (UTF-16)	\Uxxxxxxxx

By: Mhmd Darwish

Data Types

الـ Data types هي أنواع المتغيرات اللي بيتم تعريفعها في البرنامج.

type variable = value;

الـ Data types بتنقسم إلى أربع أنواع رئيسية:

- Character data types. Ex: `char, unsigned char`
- Integer data types. Ex: `int, long`
- Floating-Point data types. Ex: `float, double`
- Boolean data types. Ex: `bool`

By: Mhmd Darwish

Data Types
Numerical Types

الـ Data types هي أنواع المتغيرات اللي بنعرفها في البرنامج.

المتغيرات العددية بتنقسم لجزأين أساسيين هما الأعداد الصحيحة وتعرف بـ int والأعـداد التي تحتوي على فاصلة عشرية وتعرف بـ float.

ولكن بيتفرغ منهم Types تانية بتختلف حسب المكان اللي بتحجزه في الـ Memory.

أمثلة:

- **short int** → ودة بينصح بيه للأرقام الصغيرة لإنه بيحجز مكان أصغر
- **Long int** → ودة بيستخدم للأرقام الكبيرة زي الأرقام العلمية وبيحجز مكان أكبر
- **double** <== زي float ولكنه بيقبل أرقام أكبر وبيحجز مساحة أكبر.

By: Mhmd Darwish

Numerical Constants

الثوابت هي قيم ثابتة لا تتغير طول مسار البرنامج (read only) ولا يمكن التعديل عليها.

Form:

```
const type variable = value ;
```

Example:

```
const float pi = 3.14;
```

كدة المتغير pi احتوى على قيمة ثابتة 3.14 مستحيل مبتتغيرش .. عشان تتأكد.., اكتب حاول تدي المتغير pi قيمة تانية هيطلعلك error وش ومش بعيد يشتمك ☺

By: Mhmd Darwish

#define directive

يستخدم #define لتعريف ثابت constant أو تعريف دالة function.

- تعريف الثوابت constants:

```
#define cons value
```

مثال: لاحظ: لايوجد ; في نهاية الأمر.

```
#define PI 3.14
```

- تعريف الدوال functions:

```
#define fun(par1, par2,..) expression
```

مثال: عايزين نعرف دالة بتجمع رقمين A, B.

```
#define sub(A, B) A-B
```


By: Mhmd Darwish

Data Types
char type

من معناه كدة بيستخدم لتمثيل حرف أو رقم أو رمز صغير وبيستعمل بكثرة نظراً إنه بيحجز مساحة قليلة جداً في الذاكرة 1 byte بس.

Form:

```
char var_name = 'A' ;
```

لاحظ : قيمة المتغير اللي من النوع char بتوضع داخل علامة تنصيص مفردة.

دة اللي خدناه لحد دلوقتي .. فيه حاجات تانية كتير في الجزئية دي زي أنواع الـ Char وتمثيل الرموز طبقا لنظام ASCII (ابحث في النت ☺)

By: Mhmd Darwish

قواعد تسمية المتغيرات

الشروط الإجبارية لتسمية المتغير.

- أول حرف من اسم التغير لابد أن يبدأ بحرف من الحروف الهجائية الإنجليزية أو علامة _ .. أمثلة : ... **n1, name, age, Age, date_of_birth, _blank,**
- لا يجب أن يبدأ اسم المتغير برمز أو رقم. **1name, @darwish** - - - > false
- غير مسموح بالرموز إلا الـ _ .
- يجب أن يكون اسم المتغير بعيداً عن الـ reserved words زي cout, if, ...
- اسم المتغير case sesetive يعني الـsmall غير الـ capital نهائي. Age is not like age

فيه كمان بعض الشروط اللي يستحسن انك تعملها يعني مثلا تخلي اسم المتغير بيدل على القيمة .. استخدم الـ upperCase للتسمية زي مثلاً firstName بحيث يسهل قرائته. وهكذا.

By: Mhmd Darwish

Arithmetic Operators
العمليات الحسابية

Assume variable A holds 10 and variable B holds 20, then:

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by de-numerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

Relational Operators

Assume variable A holds 10 and variable B holds 20, then:

Operator	Description	Example
==	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
=!	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
<	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
>	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
=<	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
=>	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

Logical Operators

Assume variable A holds 0 and variable B holds 1, then:

<table>
<tr><th>Operator</th><th>Description</th><th>Example</th></tr>
<tr><td>&&</td><td>Called Logical AND operator. If both the operands are non-zero, then condition becomes true.</td><td>(A && B) is false.</td></tr>
<tr><td>||</td><td>Called Logical OR Operator. If any of the two operands is non-zero, then condition becomes true.</td><td>(A || B) is true.</td></tr>
<tr><td>!</td><td>Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false.</td><td>!(A && B) is true.</td></tr>
</table>

By: Mhmd Darwish

Assignment Operators

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	C = A + B will assign value of A + B into C
=+	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand	C += A is equivalent to C = C + A
=-	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand	C -= A is equivalent to C = C - A
=*	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand	C *= A is equivalent to C = C * A
=/	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand	C /= A is equivalent to C = C / A
=%	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	C %= A is equivalent to C = C % A

By: Mhmd Darwish

C++ Arithmetic Precedence
أولوية تنفيذ العمليات الحسابية

العمليات الحسابية بتتنفذ من اليسار إلى اليمين ولكن بأولوية معينة:

١. يتم تنفيذ العمليات داخل الأقواس أولاً.

٢. حساب الأس.

٣. الضرب والقسمة.

٤. الجمع والطرح.

مثال: مطلوب حساب قيمة `3+2*5/10+2-pow(2,3)+(10%6)`

1. `10%6 = 4` تنفيذ ما بداخل القوس
2. `Pow(2,3)=8` حساب الأس
3. `2*5/10=1` الضرب والقسمة
4. `3+1+2-8+4=2` الجمع والطرح

By: Mhmd Darwish

C++ Operators Precedence
أولوية تنفيذ العمليات

لـو في عمليـة بتحتـوي علـى عمليـات حسـابية ومنطقيـة ومقارنـة بيكـون ترتيـب تنفيـذ العمليات كالآتي:

١. العمليات الحسابية.

٢. عمليات المقارنة.

٣. العمليات المنطقية.

مثال : `2*4+3 < 6 || 3+(3%2) >2`

1. `2+4*3=12 , 3+(3%2)=4`
2. `12 < 6` → `0 (False) ,`
   `4 > 2` → `1(True)`
3. `0 || 1` → `1`

By: Mhmd Darwish

Difference between x++ and ++x

```
void main() {
    int x=10;
    int y = 3 + x++;
    cout << y << endl;
}
```

```
void main() {
    int x=10;
    int y = 3 + ++x;
    cout << y << endl;
}
```

After compiling:

x = 11, y = 13

لما ++ تتكتب بعد x :

- في الحالة دي بيحسب 11 + 3 ويحط القيمة في y.

- قيمة x بتزيد بمقدار واحد وتبقى 11.

After compiling:

x = 11, y = 14

لما ++ تتكتب قبل x :

- في الحالة بيزود واحد على x وبعدين يحسب قيمة y.

$$y = 3 + 1 + 10 = 14$$

-- كذلك الحال مع

By: Mhmd Darwish

`cin` with strings

لإدخال سلسلة حرفية عن طريق الكيبورد يتم استخدام الأمر:

```
getline(cin, Vname);
```

ويتم استخدام الهيدر `<string>`

مثال: مطلوب كتابة برنامج يكتب فيه المستخدم اسمه كاملاً.

```
#include <iostream>
#include <string>
using namespace std;

void main() {
    string name;
    cout << "Enter your full mname: ";
    getline(cin, name);
    cout << "Hello, " << name << endl;
}
```

Output

```
Enter your full mname: Mhmd Darwish
Hello, Mhmd Darwish
```


By: Mhmd Darwish

setw(n) Function
Set the *field* width

تستخدم هذه الدالة لتنسيق الخرج بحيث إنها بتتحكم في عرض العنصر (عـدد المسـافات اللي بيحجزها الـ element جوة الـ **cout**).

لاحظ: لتفعيل الدالة يتم استخدام الهيدر **<iomainp>**.

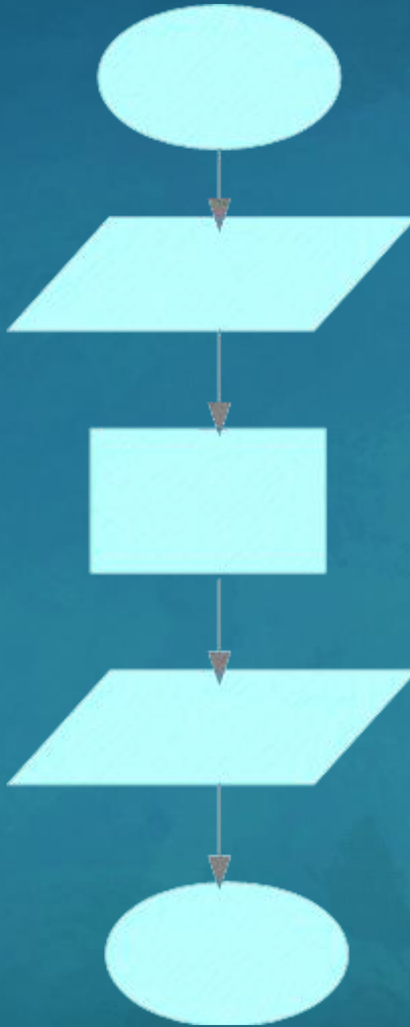
مثال يوضح الفكرة : مطلوب كتابة برنامج يطبع ٣ أرقام كل رقم يحجز ١٠ مسافات.

```
#include <iostream>
#include <iomanip>
using namespace std;

void main() {
    int a = 3, b = 4, c = 5;
    cout << setw(10) << "a" << setw(10) << "b" << setw(10) << "c" << endl;
    cout << setw(10) << a << setw(10) << b << setw(10) << c << endl;
}
```

```
         a         b         c
         3         4         5
```

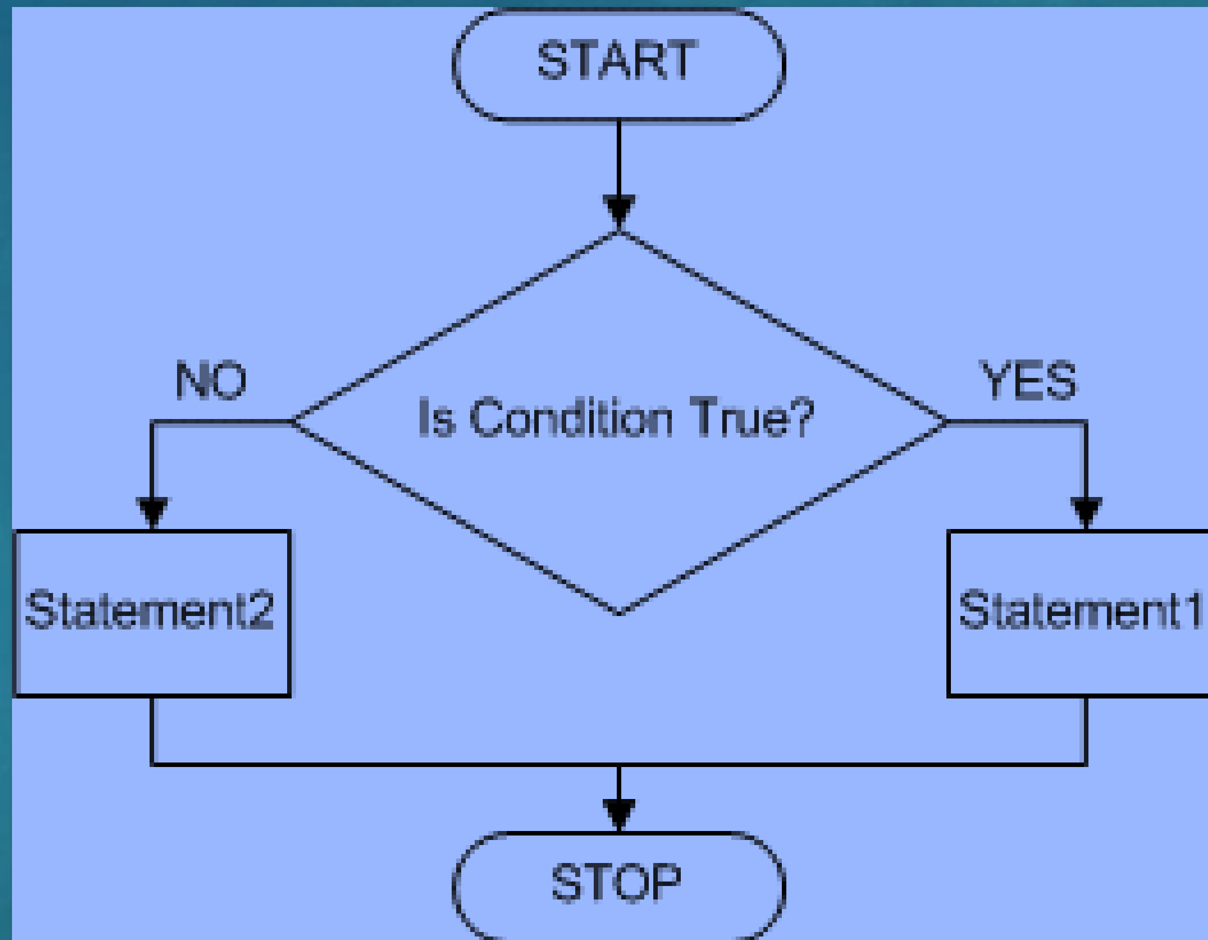
Program Structures
Sequence



البرنامج يسير في خط تتابعي بدون تفرعات

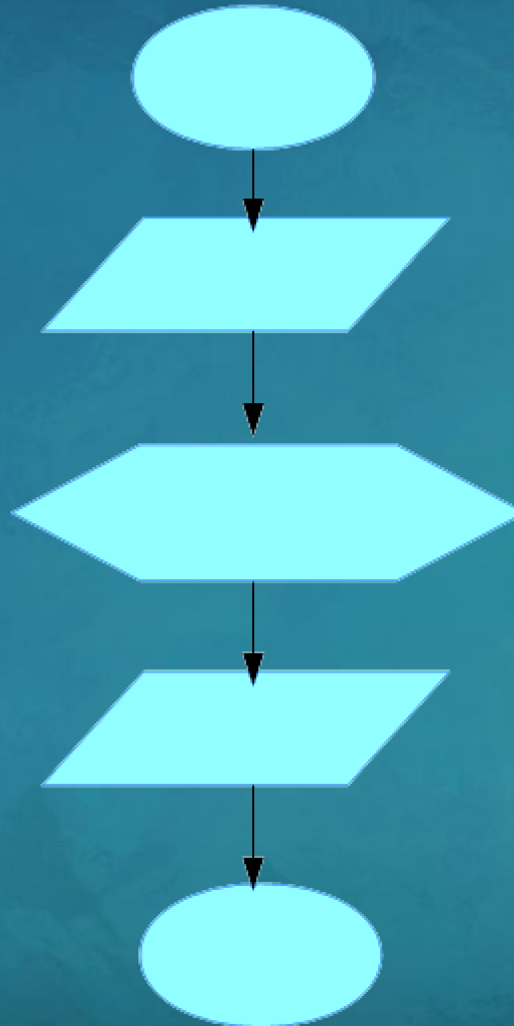
By: Mhmd Darwish

Program Structures
Branching (Condition)



By: Mhmd Darwish

Program Structures
Repetition (Loop)



By: Mhmd Darwish

Conditions
if..else Statement

Form (Syntax):

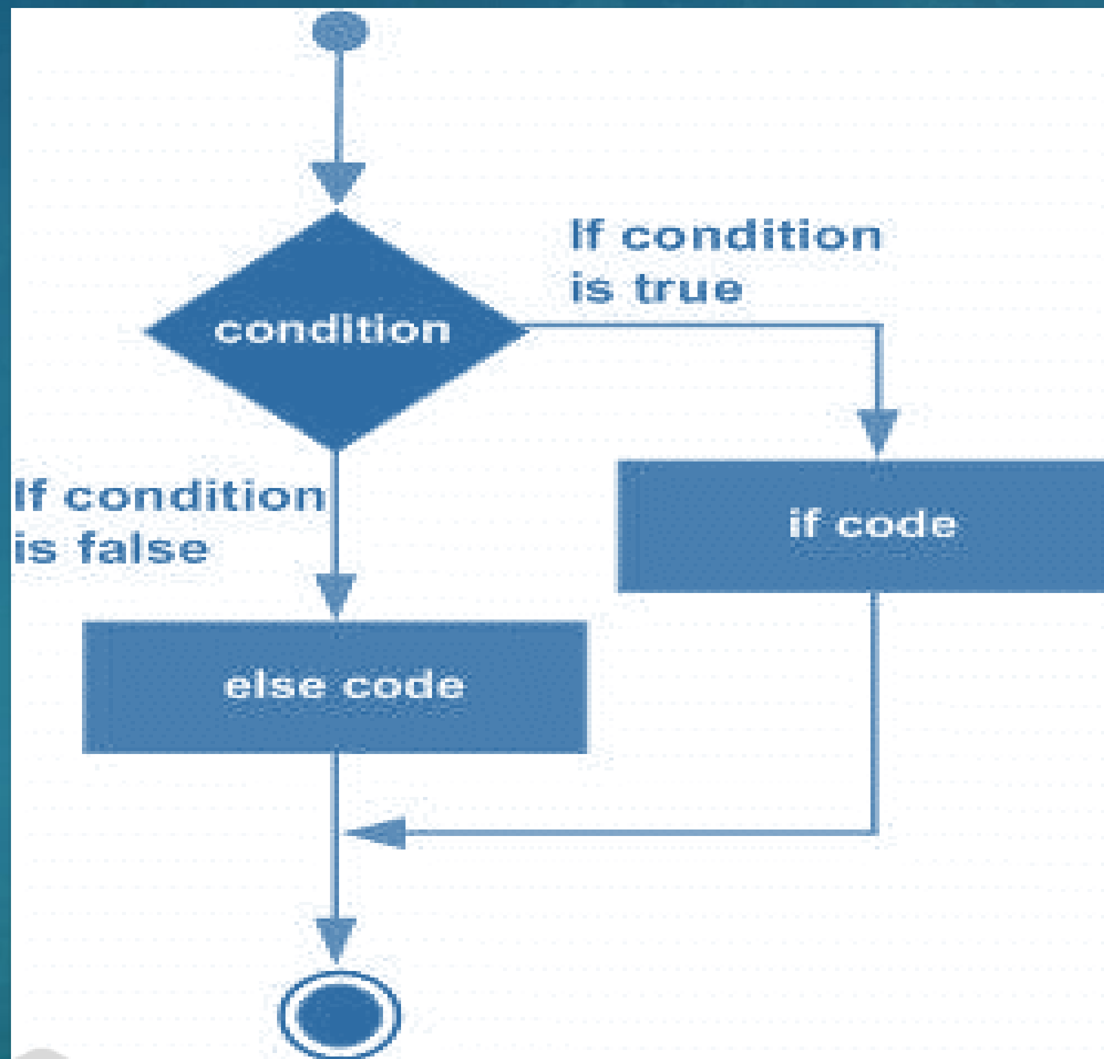
```
if(boolean_expression)
{
   // statement(s) will execute if the boolean expression is true
}
else
{
  // statement(s) will execute if the boolean expression is false
}
```

تستخدم الدالة if للتحقق من شرط أو تعبير Expression:

- لو الشرط اتحقق True بتنفذ مجموعة من السطور.
- لو Falseمتحققش بتنفذ مجموعة تانية من السطور بتتكتب بعد **else**.

By: Mhmd Darwish

Flow Diagram



By: Mhmd Darwish

else if

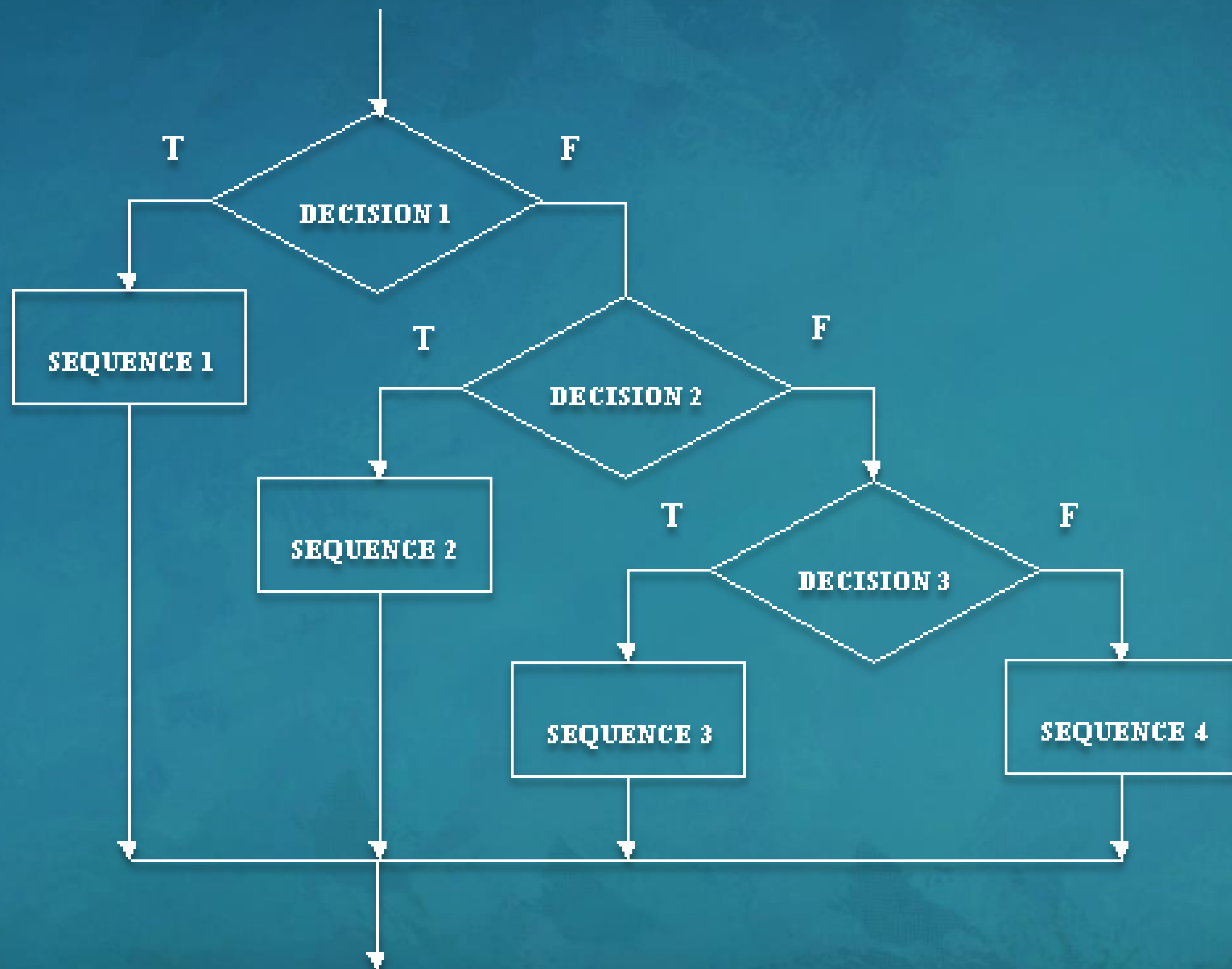
تستخدم **else if** لو عايزين نتحقق من أكتر من شرط.

في الحالة دي الشرط اللي بعد else في الآخر بيتنفذ لما ميـتحققش أي مـن الشروط الـلي فوق.

```
if(boolean_expression 1)
{
   // Executes when the boolean expression 1 is true
}
else if( boolean_expression 2)
{
   // Executes when the boolean expression 2 is true
}
else if( boolean_expression 3)
{
   // Executes when the boolean expression 3 is true
}
else
{
   // executes when the none of the above condition is true.
}
```


By: Mhmd Darwish

`else if` Flow Diagram



By: Mhmd Darwish

Nested if

Nested يعني متداخل وبالتالي بمعناها كدة هي عبارة عن if (فرعية) جوة if رئيسية.

المثال دة هيوضح الموضوع أوي: مطلوب كتابة برنامج يطبع الأكبر بين رقمين بشرط إن الرقمين أكبر من الصفر (موجبين).

```
int main() {
    int n1, n2;
    cout << "Enter two numbers: ";
    cin >> n1 >> n2;
    if (n1 > 0 && n2 > 0) {
        if (n1 > n2) {cout << n1 << " is bigger \n";}
        else {cout << n2 << " is bigger \n";}
    }
    else
        {cout << "You must enter numbers bigger than 0 \n";}
}
```

switch Statement

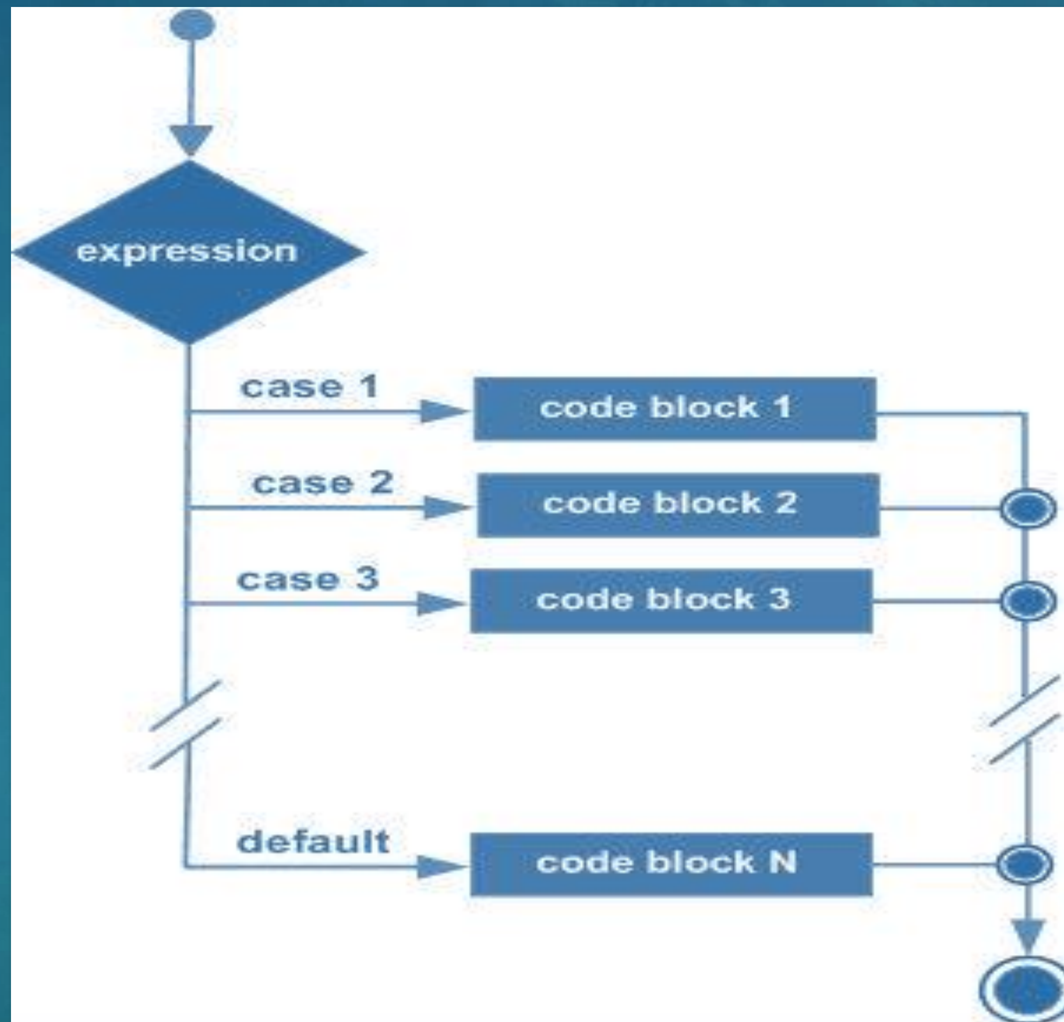
تستخدم switch للتحقق من قيمة متغير بمساواته بمجموعة من القيم. كل قيمة تسمى case ويتم التحقق من مساواة قيمة المتغير في كل case.

Syntax:

```
switch (n) {
case constant1: // must have an integer value
    //code/s to be executed if n equals to constant1;
    break;
case constant2: // must have an integer value
    //code/s to be executed if n equals to constant2;
    break;
    .
    .
default:
    //code/s to be executed if n doesn't match to any cases;
}
```


By: Mhmd Darwish

Flow Diagram



By: Mhmd Darwish

switch Notes

- قيمة التعبير اللي بعد **case** لازم تكون من النوع الصحيح integral أو enumerated زي char.
- يمكنك إدراج أي عدد من الـ cases داخل الـ **switch**.
- يتم وضع الرمز : بعد **case constant**.
- يتم وضع الـ keyword **break** بعد كتابة الـ case عشان يخرج من الـ**switch** وميتحققش من باقي القيم.
- لو مكتبتش **break** الـswitch مش هتقف والبرنامج هيخليه يتحقق من كل case لحد ما يلاقي **break** أو الدالة تنتهي.
- يتم كتابة الـ case **default** ودي اللي بتتنفذ لما كل الـ cases تكون مش متحققة.

By: Mhmd Darwish

Example on switch

```
main() {
    int n;
    cout << "Enter a number netween 1 and 3: ";
    cin >> n;
    switch (n) {
        case 1:
            cout << "You entered 1 (one) \n"; break;
        case 2:
            cout << "You entered 2 (two) \n"; break;
        case 3:
            cout << "You entered 3 (three) \n"; break;
        default: // if none of above cases is true
            cout << "You didn't enter a number between 1 and 3 \n";
    }
}
```


By: Mhmd Darwish

for Loop

الدالة التكرارية for.. ودي دالة مهمة جدًا تستخدم لتكرار أمر ما عدد من المرات طبقاً لشرط معين.

Syntax:

```
for ( init; condition; increment )
{
   //statement(s);
}
```

Conditional ? : Operator

ودي حالة شرطية مختصرة بديلة لـ if .. else.

صورتها العامة:

```
condition ? exp1 : exp2
```

exp1 لما قيمة الـ condition تكون True

exp2 لما قيمة الـ condition تكون False

وبكدة بتكون بديلة لـ if… else:

```
if (condition) {exp1}
else {exp2}
```


By: Mhmd Darwish

Conditional ? : Example

```
#include <iostream>
using namespace std;

int main ()
{
   // Local variable declaration:
   int x, y = 10;

   x = (y < 10) ? 30 : 40;

   cout << "value of x: " << x << endl;

   return 0;
}
```

```
value of x: 40
```

يعني ايه ;x = (y < 10) ? 30 : 40 ؟

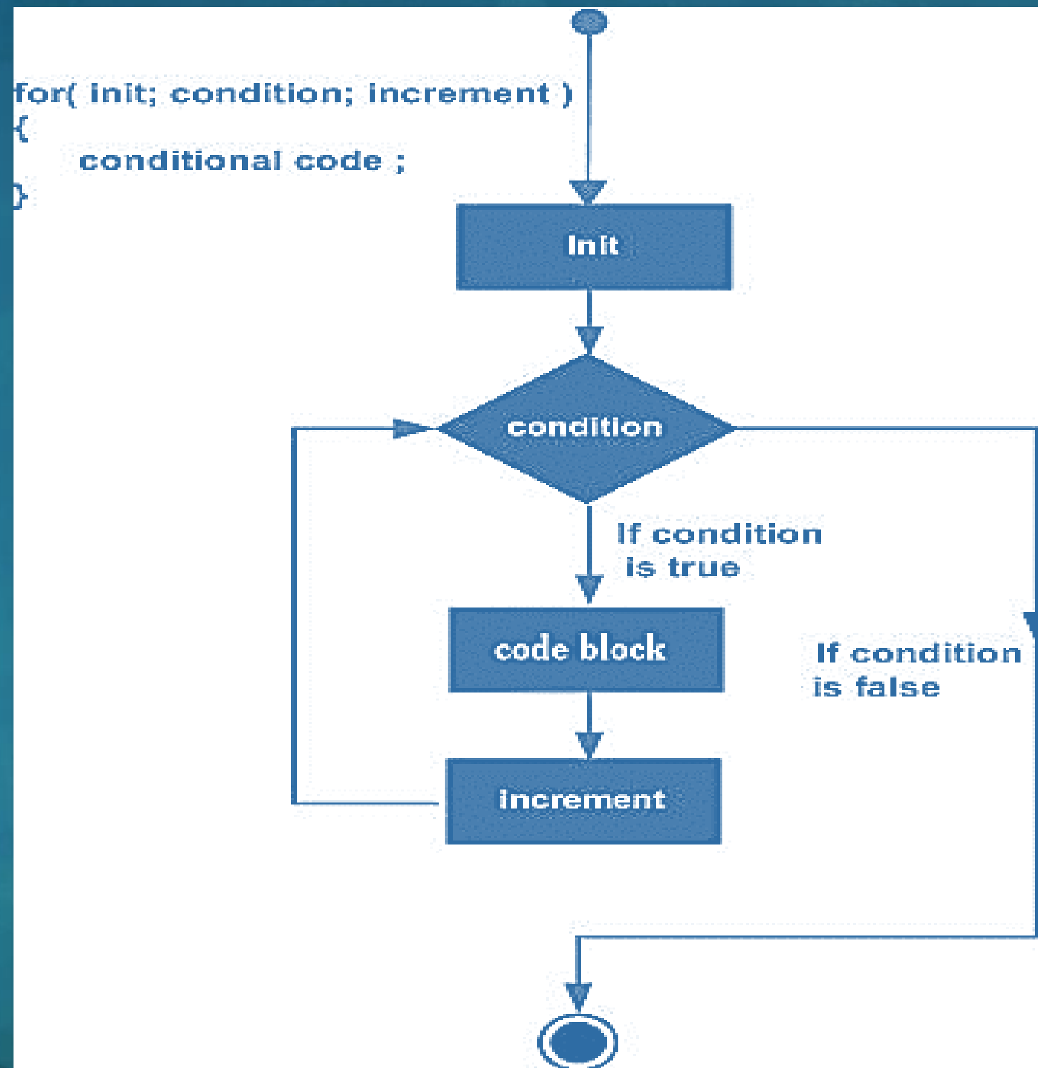
يعني هو بيعمل check .. هل قيمة y < 10 (لا) .. لو صح هيساوي قيمة x بـ 30 ولو غلط هيساوي قيمة x بـ 40 .. وبما إنها غلط هيساوي قيمة x بـ 40.

By: Mhmd Darwish

for Notes

- يــتم تنفيــذ الــ الخطــوة الابتدائيــة init أولاً ولمــرة واحــدة.. الخطوة دي بتخليك تعرّف المتغير المتحكم في الـ loop.
- ثانياً: يتم تعريــف الشرط condition وطــول مــا الشرط دة صحيح هيتم تنفيذ الـ body بتاع الــ for لحــد مــا قيمــة الــ condition تكون False.
- ثالثًا: الخطوة Increment ودة المقدار اللي هتتغير بيه قيمــة الـ init.

for Flow Diagram



COLLEGETanta
م الآخر

By: Mhmd Darwish

for Example

برنامج يقوم بطباعة الأرقام الصحيحة من ١ إلى ١٠.

```
#include <iostream>
using namespace std;

main () {
     int i = 1;
     for (i; i <= 10; i++)
     {cout << i << endl;}
}
```

```
1
2
3
4
5
6
7
8
9
10
```


By: Mhmd Darwish

while Loop

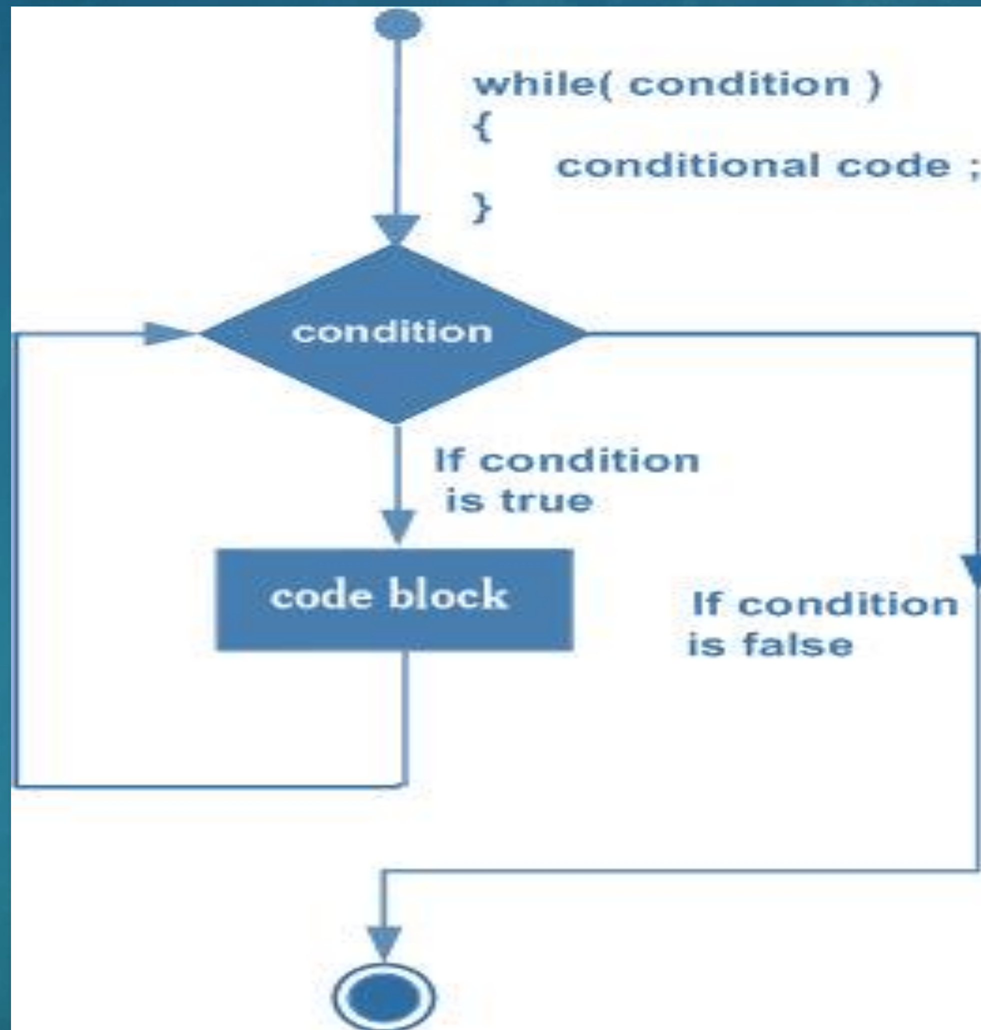
تستخدم الدالة while لتنفيذ مجموعة من الأوامر تبعاً لشرط معين (طول ما الشرط صحيح الدالة بتكرر نفسها). لحد ما الشرط يبقى False .. في الحالة دي بيطلع من while.

Syntax:

```
while(condition)
{
    //statement(s);
}
```


By: Mhmd Darwish

while Flow Diagram



By: Mhmd Darwish

while Example

برنامج يقوم بطباعة الأرقام من ١ إلى ٢٠ بزيادة ٥ أرقام.

```
#include <iostream>
using namespace std;

main () {
    int i=1;
    while (i <= 20) {
        cout << i << endl;
        i += 5;
    }
}
```

```
1
6
11
16
```


By: Mhmd Darwish

do.. while

do.. while شبيهة بـ while ولكن إيه الفرق:

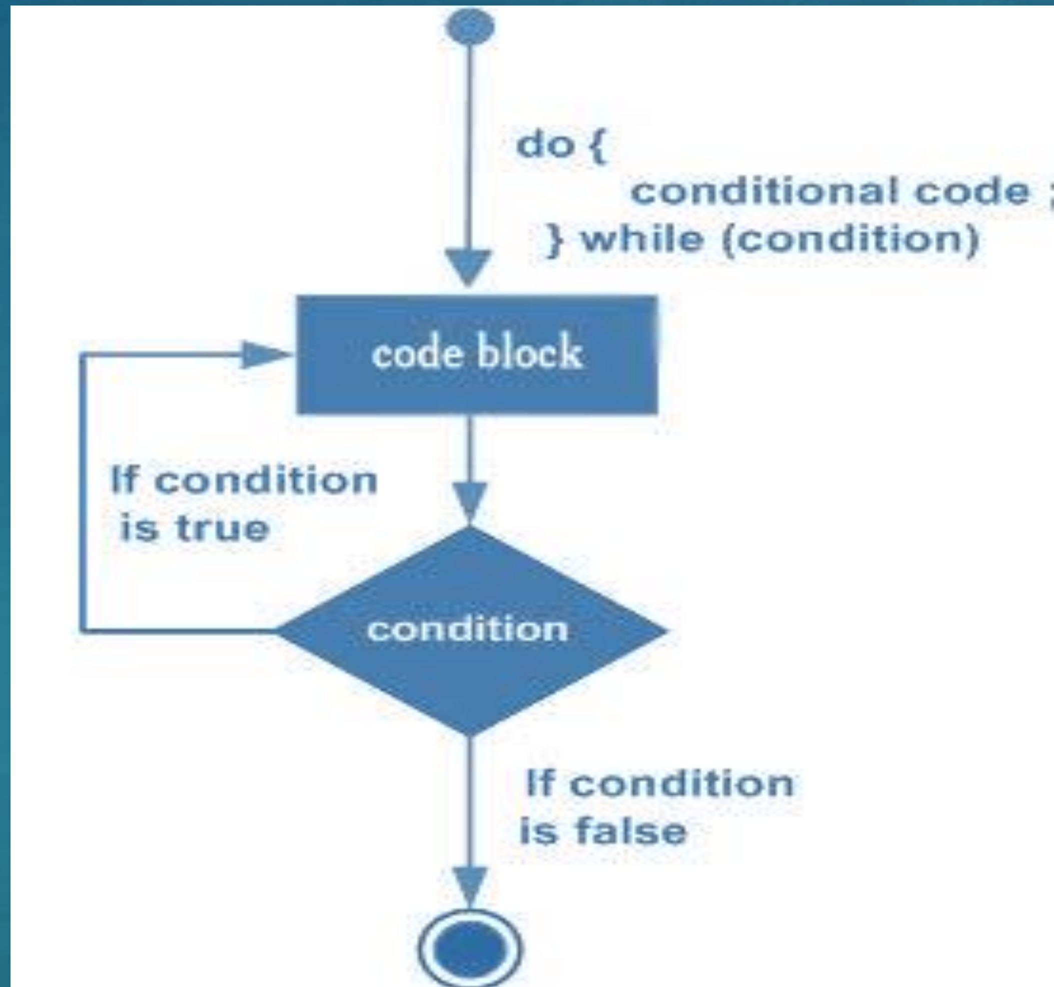
do..while بتنفذ الـ statements بعد do مرة واحدة على الأقل قبل ما يتحقق من الشرط المكتوب بعد while.

Syntax:

```
do
{
    //statement(s);
}
while( condition );
```


By: Mhmd Darwish

do.. while Flow Diagram



By: Mhmd Darwish

do.. while Example

```
#include <iostream>
using namespace std;

main ()
{
    // Local variable declaration:
    int a = 10;

    // do loop execution
    do {
        cout << a << endl;
        a++;
    }
    while( a <= 20 );
}
```

```
10
11
12
13
14
15
16
17
18
19
20
```


By: Mhmd Darwish